GSOC 2015 PROPOSAL TO P2PSP.ORG

PROJECT: INTEGRATION OF GUI,PLAYER AND PEER

PRINCE

March 24, 2015

1.What project would you like to work on and why?

I am going to design and implement the graphical user interface for the Peer in the P2PSP python implementation.Player will be also integrated with the GUI.

These are the reasons why I want to work on this project:

- I enjoy developing interactive softwares.
- I will have an opportunity to learn more about pygame and libvlc framework.
- I am confident about completing this project within GSOC period successfully.

To prove that I am completely capable of completing this project successfully I have also created a demo-p2psp-gui on github.

URL: https://github.com/maniotrix/demo-p2psp-gui

Working of demo version can be also viewed on youtube: demo_p2psp_gui

URL : https://www.youtube.com/watch?v=BUIIM5pl_hQ

2.Propose an idea of how are you going to develop the project.

Project Detailed Description

- Application will be structured using the Model-View-Controller(MVC) pattern.

The main aim of this project is to control peers and their configuration ,control player,control peer and player communication . Hence interacting with peers and player and coordinating with them through GUI.

Model Component:

- **Each class will have getters and setters for their properties.This will help in accessing and altering the properties easily.Specially for modifying data in p2psp_peer and category class which is mentioned below.**
- **Classes,properties and methods - their number or their name, mentioned in below API are not fixed.Modules may be removed,added,modified during development process. Each and every commitment to API will be discussed with mentor and community.**
- **Widget modules will be also created for different widgets.**

#Modules(API):

- pygame_surface_builder: This module manages(initialization and vice-versa) pygame environment and provides a surface/screen.
    - properties: screen,window_id
    - methods:
        - get_window_id: returns window id of the screen.
        - methods for controlling pygame.mixer .
- player: The module with Player class is responsible to play stream using libvlc framework.Only one object of this class will be used throughout the application.
    - properties: Vlc_Instance, player
    - methods:
        - get_vlc_instance: returns vlc instance.
        - get_player : returns vlc MediaPlayer object.
        - set_window: sets the window_id where media will be played.
        - set_stream_mrl: sets the mrl of the stream to be played.
        - play,pause and stop player.
- identity: module contains Identity class.

- ➢ properties: ID,name
- ➢ methods: getters and setters for each property.

➔ Below classes when developed will inherit from identity module.

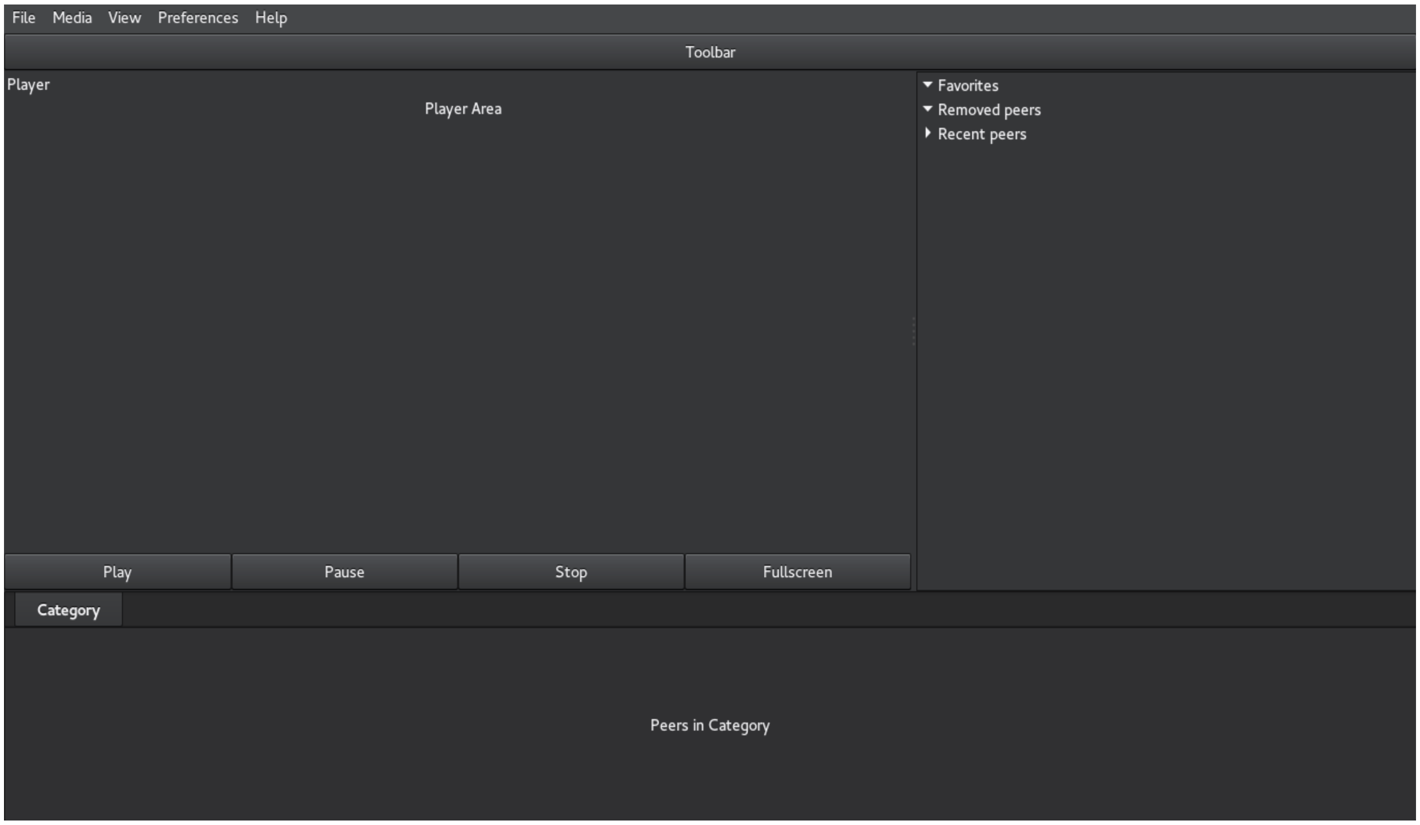
- ❖ p2psp_peer: This will contain a P2psp_Peer class for storing data of different peers.Objects of this class have their own set of properties to get configured separately .This is necessary as peer parameters which are available from peer modules like Peer_IMS are static variables.
  - ➢ Peer Class:
    - ■ parameters: all the peer parameters from P2PSP Project.
- ❖ category: Module with Category class acts as a container for storing peers.Each category will maintain a list of peers.
- ❖ peer_session: Will have a class Peer_Session .Only module which will import peer module from P2PSP Project.This class will also inherit from threading module.Thus peer.Peer(from p2psp project) object will be created in a new thread.

➔ API will contain few utility modules like File I/O ,parsing etc.

View Component:

#GUI

Using my experience of developing an interactive multi threaded download manager and after experimenting and reading peer modules in P2P2SP Project, I have decided to go through the following GUI:



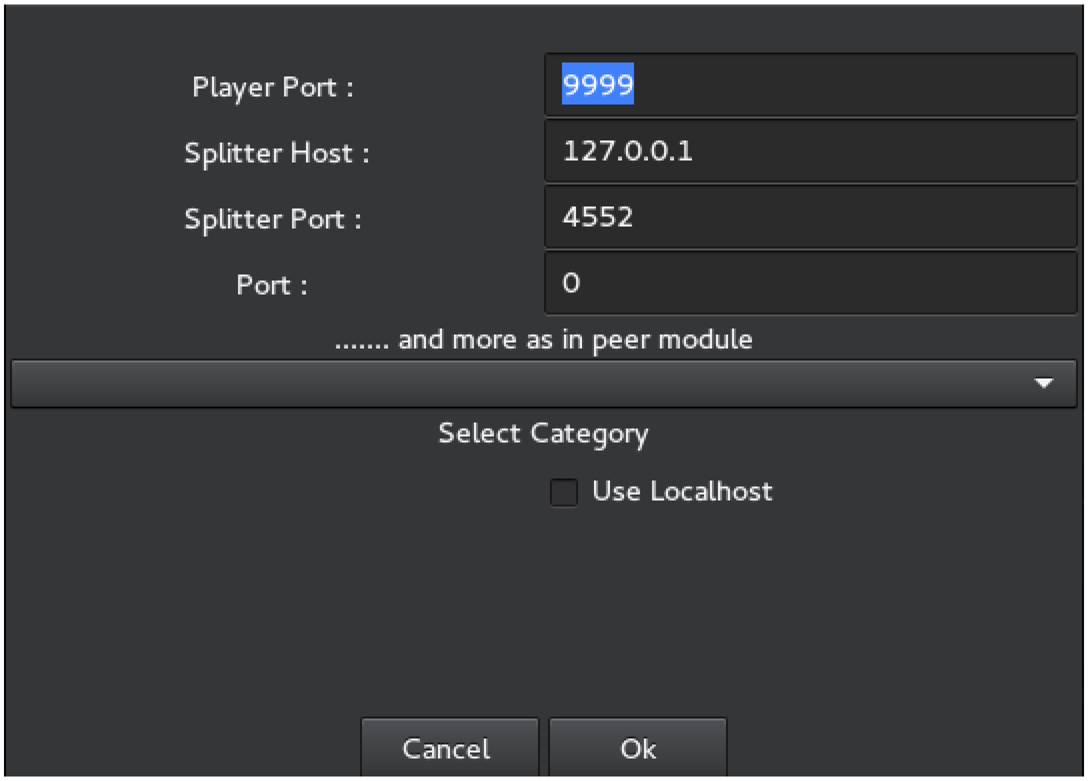
As shown in above figure these are different parts of GUI.Below is the detail of each and every part I am going to implement.

GUI Design:

Each part is implemented individually i.e. each part is an independent class.Each class could be used independent of each other.And at the end each part will be connected as required.This will not only help in programming but will also make source code much easier to read and understand.Moreover, it will reduce bugs and will help in finding the cause of bugs easily.

- ➔ All the GUI components will be made using glade. Further modifications in GUI may occur during development process.
- ➔ Buffer_Status and Number_of_peers_in_team will not be displayed for every individual peer in the list from 'TreeView' in notebook. These two parameters will be displayed only when peer plays the stream in the Status Bar.

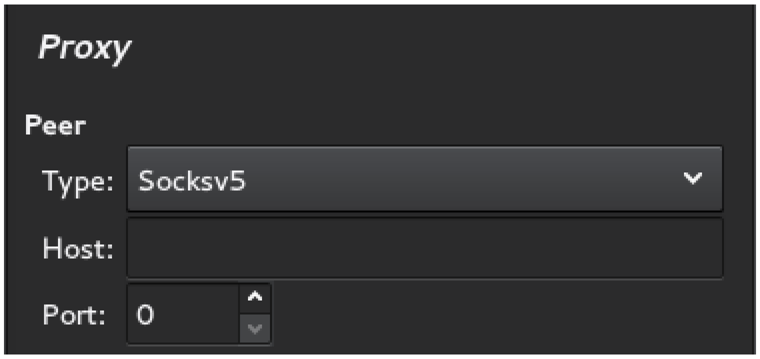
- Main Window: All the parts Menu Bar , Toolbar , Drawing Area (surface for media), Notebook widget (Tabbed windows) for peers list , Tree Views(properly configured peers list) , list frequently(favorite) used peers, list of removed peers.
- Menu Bar:
  - File Menu:
    - New: Creates a new peer.When this menu item is clicked a dialog box as shown in below figure is opened .After that the peer is added to the specified category in the notebook widget.There will be an option to create a new category for the peer.



Player Port : 9999
Splitter Host : 127.0.0.1
Splitter Port : 4552
Port : 0
.... and more as in peer module
Select Category
Use Localhost

    - Import : This will import an existing file which consists of one or more categories. After that a dialog box will appear which lists all the peers in each category.On pressing ok categories will be added to notebook.
    - Export : Existing peers from each category will be displayed in a dialog box and then written to a json file on pressing ok.
    - Close : quits application.
  - Media Menu:
    - Play: Command to play currently selected peer
    - Pause: Command to pause currently selected peer
    - Stop: Command to stop currently selected peer
    - Restart: Command to restart currently selected peer
    - Remove: Command to remove currently selected peer
  - Preferences Menu:

- Proxy Settings: Configure proxy as per the network as shown in figure.Authentication will be also implemented.



Proxy

Peer

Type: Socksv5

Host:

Port: 0

- more to come······

➢ View Menu:
- Toggle toolbar
- Toggle Notebook
- Toggle Side Widgets
- Fullscreen Player
- Language Selection(If Implemented)

➢ Help Menu:
- Peer Manual:
- Check for Updates
- About

❖ Tool Bar: This widget will contain buttons for menu options.Buttons will have images on them.

❖ Surface(Player Area): surface developed using pygame to display media stream.

❖ Scrolled Window:

➢ Favorite Peers : Expander widget to display favorite peers

➢ Removed Peers: Expander widget to display removed peers.

➢ Recent Peers : Expander widget to display recent peers.

❖ Scrolled Window:

➢ Notebook(Peers in Category): Switching between different category tabs .Here at least one tab for displaying all peers will be provided by default.Inside the notebook peers can be viewed as list or grid of icons.
- Treeview:contains list of peers in the category. On double clicking the mouse on the current selected peer , player will play the stream associated with the peer.

❖ Status Bar: displayed at the end of Main Window.

➢ currently playing peer

- Buffer_Status of currently playing peer.
- Number_of_peers_in_team in currently playing peer.
- download/upload status

- Popup Menu:This menu will appear inside TreeView on Mouse_Right_Click at selected peer.
  - contains options available in 'Media Menu', edit selected peer.

Controller Component:

#controls

- Event Handlers will be available for each and every widget in GUI. Proper accelerators will be provided for some widgets.

➔ Two handlers used most by users will be:

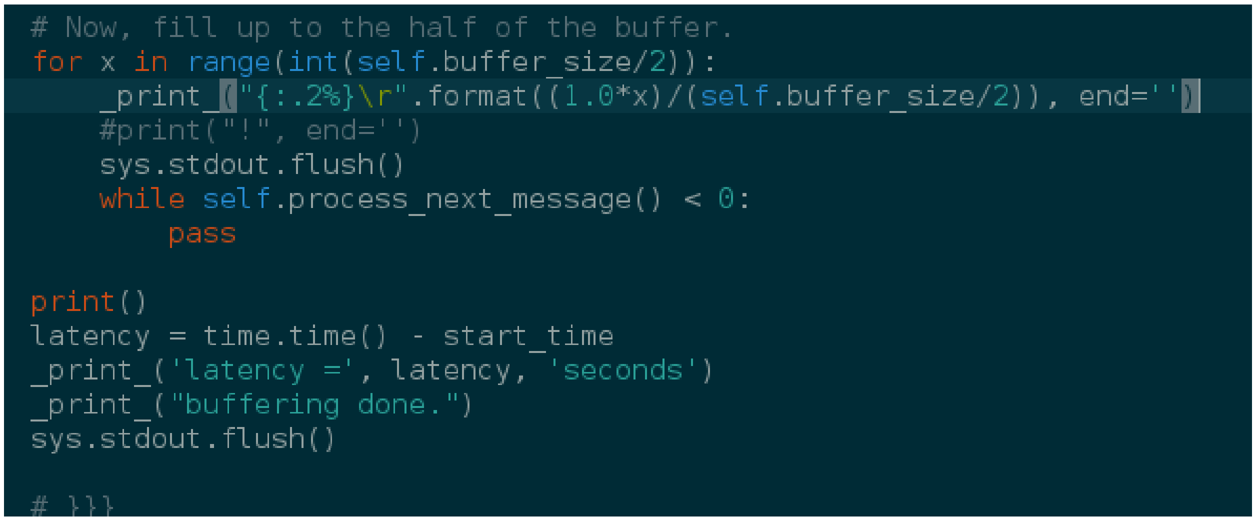
- create_new_peer:
  - As described in 'New Menu_Item' ,after ok button is clicked followings steps will take place:
    - a new p2psp_peer object will be created.
    - object will be added to respective lists of 'All' and 'Recent' objects.
    - new category object will be created if user specifies different category besides which are available.
    - next peers will be displayed in notebook.
- play_selected_peer:
  - After peer is added to 'TreeView' in Notebook widget, peer can play stream according to its configuration.Steps taken when a peer is double_clicked:
    - Player's mrl is set according to peer configuration.
    - All the static variables from peer modules in P2PSP Project are modified accordingly.A new Peer_Session object(thread) will be created.
    - Player's play method called to display stream.
- stop_player:
  - Player's stop method is called.
  - Peer_Session automatically dies when Player is disconnected.

- refresh_peer:
  - stop_player is called.
  - play_selected_peer is called.

➔ **In the above components , two important parameters Buffer_Status and Number_of_peers_in_team were not discussed.**

- So how to get these parameters to be displayed on GUI?
  - After going through source code of p2psp ,I found these parameters in Peer_IMS and Peer_DBS in following piece of code.

Peer_IMS ->

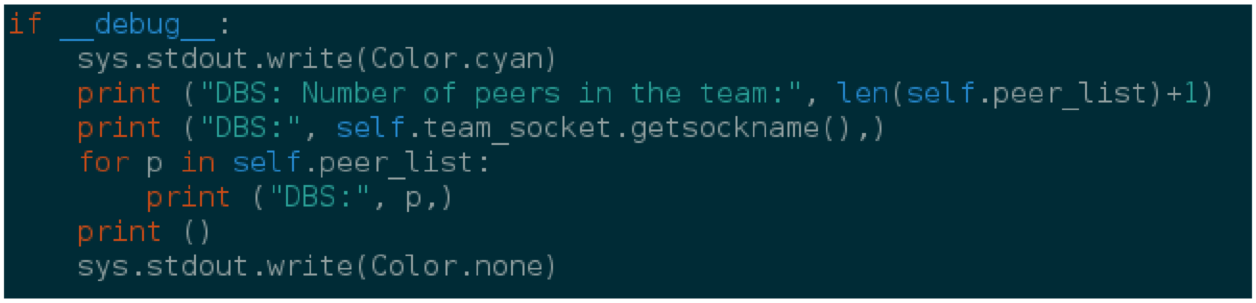


```
# Now, fill up to the half of the buffer.
for x in range(int(self.buffer_size/2)):
    _print_("{:.2%}\r".format((1.0*x)/(self.buffer_size/2)), end='')
    #print("!", end='')
    sys.stdout.flush()
    while self.process_next_message() < 0:
        pass

print()
latency = time.time() - start_time
_print_('latency =', latency, 'seconds')
_print_("buffering done.")
sys.stdout.flush()

# }}}
```

Peer_DBS ->

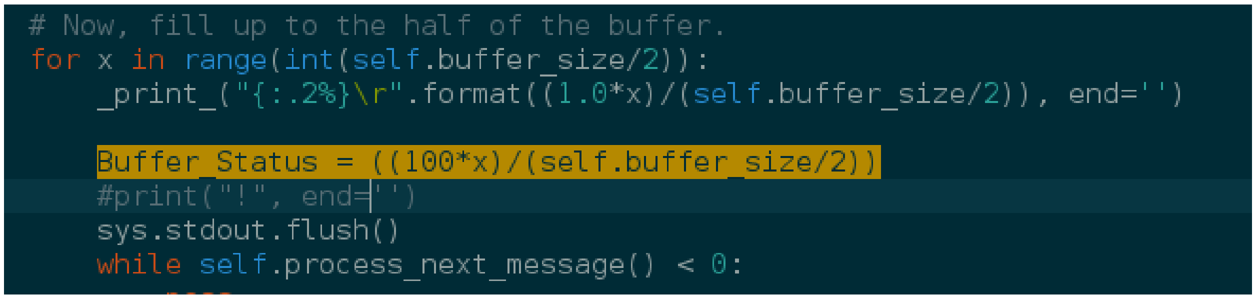


```
if __debug__:
    sys.stdout.write(Color.cyan)
    print ("DBS: Number of peers in the team:", len(self.peer_list)+1)
    print ("DBS:", self.team_socket.getsockname(),)
    for p in self.peer_list:
        print ("DBS:", p,)
    print ()
    sys.stdout.write(Color.none)
```

We have one player in our GUI. So at a time only one Peer_Seesion() object(thread) will be running in the application.

Therefore what I propose is that there should be a global or static variable in each of the modules to get these parameters.

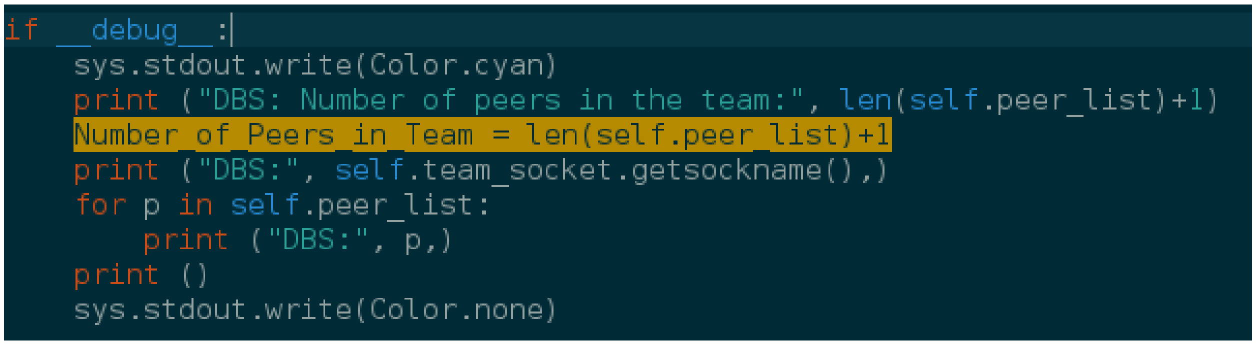
Following code snippet show the modifications.
Peer_IMS ->



```
# Now, fill up to the half of the buffer.
for x in range(int(self.buffer_size/2)):
    _print_("{:.2%}\r".format((1.0*x)/(self.buffer_size/2)), end='')

    Buffer_Status = ((100*x)/(self.buffer_size/2))
    #print("!", end='')
    sys.stdout.flush()
    while self.process_next_message() < 0:
```

Peer_DBS ->



```
if __debug__:
    sys.stdout.write(Color.cyan)
    print ("DBS: Number of peers in the team:", len(self.peer_list)+1)
    Number_of_Peers_in_Team = len(self.peer_list)+1
    print ("DBS:", self.team_socket.getsockname(),)
    for p in self.peer_list:
        print ("DBS:", p,)
    print ()
    sys.stdout.write(Color.none)
```

Apart from what discussed above this software can be also extended to include some features like **network availability,software persistence and state management** and many more as we can propose.

How the application's state will be managed?
I propose to accomplish this goal using **pickle** module.

I was also thinking about **integrating (splitter+peer) in P2PSP GUI** (could be done within GSOC period or after that). I discussed this idea with project's mentor Cristóbal . The discussion concluded in discussing this idea on the mailing list to know the community opinion.

If time permits(which I think will) this application could be developed into a **multilingual software.**

Timeline(May 25 to AUG 24): Based on the above parts after going through events and timeline page at google-melange, I have created deadlines for myself to complete my work.Hence, I will try my best to complete each part.

These Periods and Phases are:
Community Bonding Period(April 27 to May 25): In this period, I will assemble all prerequisites related to coding. Actually what I think is that one should have a clear understanding of the source code and implementation of P2PSP protocol, for developing an awesome and user-friendly interactive software which would help P2PSP project. This will include taking guidance from my mentor. In this period, if my mentor thinks that I should start programming the project then I will start doing. Early completion will help in completion of the project.

1. **First Coding Period(May 25 to Midterm evaluations):** Within this period I will cover implementation of API for Model and View Components, each and every component of GUI will be finalised,handlers for widgets will be created and each and every feature of player will be implemented. Actually upto this period the application will be properly functional except following features:

1) right side panel for expander widgets will not be functional(though already have been part of GUI).
2) "Preferences" menu item will have no item at this point.Therefore no application as well as network settings will be available.
3) Application's state management would not have been handled by this time.

25 May to 31 May: I will implement modules from both Model And View Components except those involving pygame and libvlc.
1 to 7 June: Player and pygame_surface_builder modules will be implemented. All the player features like play ,stop ,pause, fullscreen , volume configuration and Key bindings for these features will be also implemented.
8 to 25 June: Handlers for each widget will be implemented and connected to their respective widgets.
At this position any stream provide by peer to player has to be properly played along with all the player features.User should be able to switch between different peers dynamically.

26 to 3 July: This is the midterm-evaluation period. So my mentor will evaluate my progress so far. All the project status so far and future will be discussed. Any bug in the application remaining so far will be also fixed.

2.After Midterm Evaluations(4 July) to suggested Pencils down date(16 Aug) : All the features which were pointed out not to be developed upto mid term Evaluation period will be properly implemented.If time permits multilingual features will be also implemented.

3. After suggested to firm pencils down date : Proper tests for handlers will be implemented to test them rigorously and find any bug . Required refactoring of code will be done appropriately. Improvements in documentation will be done. If possible wiki page for project will be also created on github.

After Gsoc Period: I would like to continue maintaining my project . I have also plans for Splitter_GUI and introduce more features after Gsoc Period. I would like to make application multilingual in as many languages as suggested from community.

3.My experiences in free software development.:

I started contributing to open source when started working on my multi threaded download manager. Since then I have been regularly active on Github as user **maniotrix.**

Projects contributed on github:

- MildHop: An android game using Unity3d (C#).
- Xblunt: A multi threaded download manager(java).
- p2psp
- demo-p2psp-gui : A demo version of this project(python)

Apart from github I have also contributed in Eclipse Plugin Project Efxclipse. I have contributed fixing Bug 461121.

4.Why do you want to work with P2PSP Project in particular?

The most important reason is that P2PSP Project is Open Source and computer networks has been always a fascinating topic for me. So if I get involved in this project I get to learn a lot from the experiences of this community.This project has also an amazing and helpful community. I am looking forward to contribute a lot in this project in future.

5.Will you be working full-time on the project for the summer, or will you have other commitments too (a second job, classes, etc)?

- I do not have any other commitments. Completing this project successfully is my top priority in summer.

6.What is your ideal approach to keeping us informed of your progress, problems, and questions over the course of the project?

I will discuss everything, whether its progress,problems,questions etc.) on mailing list and irc channel.

I am assuming project will be maintained on github. If this is so my progress can be also tracked using commit messages and issues(for problems and questions). Whenever I will commit I will explain everything related with that commit.

7.How can we contact you to ask you further questions?My email addresses:

Primary: prince.mst13@itbhu.ac.in

Secondary: princek082@gmail.com

8.What school are you attending? What year are you, and what's your major/degree/focus? If you're part of a research group, which one?

The Student:

Name: Prince
e-mail : prince.mst13@itbhu.ac.in
Github : https://github.com/maniotrix

The Institute:

University: Indian Institute of Technology (BHU), Varanasi
Major: Materials Science and Technology
Year: Sophomore
Degree: Integrated Dual Degree(Btech and masters)